Tools that can help with testing the output of the simulator done for the final project of course 02155.

The tools are provided in a zip file called: test_final_proj.zip
They based on bash scripts, and they should be working on all major operating systems: Windows (by using WSL), MacOS and Linux.

The tests that we are going to do in the lab are provided as 3 different files (just like the tests that you can see on github):

- the .s file contains the test program written in assembly
- the .bin file contains the binary images that is produced after the program is compiled. This is the file that your simulator takes as input
- the .res file contains the values from the 32 general purpose registers of the RISC-V at the end of the simulation
- in some cases you might get also a .c file if the test program was written in c

So basically, to check if your simulator works OK, you'll need to compare the expected result provided with the test in the .res file with the register values that your simulator produces when the simulation of the test program is finished.

Note: When you debug your simulator, you can also check if it works OK by printing the values of registers after each instruction and then running the same program in Ripes and comparing the value of the registers from Ripes with the values that your simulator returns.

So, the purpose of these tools is to make it easy to see the values of the registers saved in a .res file, either in hexadecimal representation or in decimal representation. This is very relevant for seeing the values of the registers in the .res files provided with the test. In case you have also saved the registers at the end of the simulation in a .res file then you can use these tools to see the contents of your registers and to check if there are any differences from the expected value of the registers (available in the .res file provided with the test).

Format of the .res files
The .res files have exactly 128 bytes corresponding to the values of 32 registers (32 registers x 4 bytes/register=128 bytes). The bytes for R0 are located as the first 4 bytes in the file, the bytes for R1 are located as the next 4 bytes in the file etc.

The bytes for registers are saved in the .res file using the little-endian byte ordering. This means that the least significant byte is saved at the lowest address (of the 4 addresses used by the 32-bit value). The other bytes will follow, where the most significant byte is located as the higher address. As an example, the value: x12345678 will be stored in memory in this order: x78, x56, x34, x12 (where x78 is the least significant byte and is located at the lowest address and x12 is the most significant byte and is located at the highest address). The tools also include a conversion script that can convert between big-endian and little-endian byte ordering.

For more info about big and little endian byte ordering check out this link or eventually this link.

Installation instructions

Unzip the test_final_proj.zip file into a directory of your choosing. To avoid potential problems it's a good idea that the directory is not is not on a path that contains spaces or not-English letters (æ,ø,å,ö,ü etc…).

Then open a terminal window (in windows you can use the power shell/terminal, where you execute the wsl command). Next using the cd command go to the directory where the files where unzipped: **cd /path/to/the/directory**

To make the tools accessible from the terminal even if you change the current directory then run the command: **source 02155_install.sh**
This script will add the path to the directory where the tools for testing the final project for course 02155 are installed. The path is only added for the current terminal. If you want to use another terminal, you'll have to run again the source 02155_install.sh script.

Then using the cd command go to the directory where you have your tests.
Then run the tests, and for each test generate the output for your simulation. It's good to format the output on the terminal window so that it can be easily humanly readable where you print the value of each register like this:

x0 = 0xDEADBEEF, x1 = 0xDEADBEEF, x2 = 0xDEADBEEF, x3 = 0xDEADBEEF
x4 = 0xDEADBEEF, x5 = 0xDEADBEEF, x6 = 0xDEADBEEF, x7 = 0xDEADBEEF
..and so on.

You should also generate a binary dump of the register contents of registers x0-x31, in that order. This is basically a file with the extension ".res" that will contain the values of the general-purpose registers. Since we have exactly 32 registers the file should have exactly 128 bytes (32*4). It's best if you use little-endian byte ordering so that it's the same byte ordering as in the .res files provided with the test.

Description of the test tools for course 02155

02155_check_output.sh <correct.res> <yourfile.res>
Takes as argument two .res files:

- <correct.res> file is the .res file that comes with the test and contains the correct values that the registers should have at the end of the program.
- <yourfile.res> file is the file you have generated with the help of your simulator and contains the final values of the registers when the simulation is done.

This command mentions if the registers are correct and if not then mentions which registers are wrong and what value they were supposed to have.

02155_show_hex_nonzero.sh <filename.res>
Shows on the display only the values of the registers that have a value different from zero. The values are displayed using hexadecimal representation.

02155_show_hex.sh <filename.res>
Shows on the display the values of all the 32 registers. The values are displayed using hexadecimal representation.

02155_show_dec_nonzero.sh <filename.res>
Shows on the display only the values of the registers that have a value different from zero. The values are displayed using decimal representation.

02155_show_dec.sh <filename.res>
Shows on the display the values of all the 32 registers. The values are displayed using decimal representation.

02155_convert_endianness.sh <input_file.rec> > <output_file.rec>
Example: 02155_convert_endianness.sh test.rec > test_out.rec
The values from the .rec files should be saved using the little-endian notation. In case you've saved the files using the big-endian byte order then when you try to see the register values using for example the command 02155_show_hex.sh the values are incorrect. In that case you can fix the file by converting it to a little-endian byte order by running the command **02155_convert_endianness.sh.**

Note:
In case you would like to use a hex editor in order to see the raw content of the binary .res file then I can recommend HxD for windows, and Sublime Text for all 3 operating systems.